

# Table of Contents

<b>Pleiades.....</b>	<b>1</b>
<u>Pleiades Configuration Details.....</u>	1
<u>Preparing to Run on Pleiades Broadwell Nodes.....</u>	3
<u>Preparing to Run on Pleiades Haswell Nodes.....</u>	6
<u>Preparing to Run on Pleiades Ivy Bridge Nodes.....</u>	9
<u>Preparing to Run on Pleiades Sandy Bridge Nodes.....</u>	11
<u>Broadwell Processors.....</u>	13
<u>Haswell Processors.....</u>	15
<u>Ivy Bridge Processors.....</u>	18
<u>Sandy Bridge Processors.....</u>	20
<u>Hyperthreading.....</u>	22

# Pleiades

## Pleiades Configuration Details

### Pleiades Hardware Summary

- 153 racks
- 10,872 nodes
- 236,032 CPU cores + 184,320 GPU cores
- 893 TB total memory
- 7.24 petaflops (PF) theoretical peak performance (CPU) + 0.275 PF (GPU)

### Compute Node Hostnames

There are 4 individual rack units (IRUs) in each rack, with 18 compute nodes per IRU. The hostname of each node is based on the physical rack (r) and IRU (i) it resides in, and its node position in the IRU (n). For example: r301i2n3.

Note: A single rack leader controls two racks, so the naming convention for the 144 nodes residing in every two racks (2 racks x 4 IRUs x 18 nodes) uses only odd rack numbers.

The following naming conventions are used for the compute node hostnames:

Sandy Bridge

r[3xx]i[0-7]n[0-17], where 3xx are *odd* numbers between 305-312 and 317-330;  
r313i[0-3]n[0-15] (GPU)

Ivy Bridge

r[4xx]i[0-7]n[0-17], where 4xx are *odd* numbers between 401-472; and 481-482;  
483i[0-3]n[0-17]

Haswell

r[5xx]i[0-7]n[0-17], where 5xx are *odd* numbers between 509-516, 573-580, and  
585-596; 583i[4-7]n[0-17]

Broadwell

r[6xx]i[0-7]n[0-17], where 6xx are *odd* numbers between 601-608, and 617-636

Note: Pleiades compute nodes are accessible only through PBS jobs.

### Pleiades Processor, Memory, and Network Subsystems Statistics

The following tables provide detailed configuration statistics for each type of node.

	Node Details		
	Sandy Bridge	Ivy Bridge	Haswell
Architecture	ICE X	ICE X	ICE X
Processor	8-core Xeon E5-2670	10-core Xeon E5-2680v2	12-core Xeon E5-2680
Newest Instruction Set	AVX	AVX	AVX2
Hyperthreading	ON	ON	ON

Turbo Boost	ON	ON	ON
CPU-Clock	2.6 GHz	2.8 GHz	2.5 GHz
Maximum Double Precision Flops/Cycle/Core	8	8	16
# of Cores/Node	16	20	24
Total # of Nodes	1,512	5,256	2,052
Total # of Cores	24,192	105,120	49,248
Total Double Precision TFlops	599	2,355	1,970

### Memory

	<b>Sandy Bridge</b>	<b>Ivy Bridge</b>	<b>Haswell</b>
L1 Cache	Local to each core; Instruction cache: 32K Data cache: 32K; Associativity: 8; Cache line size: 64B	Local to each core; Instruction cache: 32K Data cache: 32K; Associativity: 8; Cache line size: 64B	Local to each core; Instruction cache: 32K Data cache: 32K; Associativity: 8; Cache line size: 64B
L2 Cache	256 KB per core; Associativity: 8; Cache line size: 64B	256 KB per core; Associativity: 8; Cache line size: 64B	256 KB per core; Associativity: 8; Cache line size: 64B
L3 Cache	20 MB shared by the 8 cores; Associativity: 20; Cache line size: 64B	25 MB shared by the 10 cores; Associativity: 20; Cache line size: 64B	30 MB shared by the 12 cores; Associativity: 20; Cache line size: 64B
TLB	Local to each core	Local to each core	Local to each core
Default Page Size	4 KB	4 KB	4 KB
Memory/Core	2 GB; DDR3	3.2 GB; DDR3	5.3 GB; DDR4
Total Memory per Node	32 GB	64 GB; 3 nodes at 128 GB	128 GB
Memory Speed and Bandwidth	1600 MHz; 4 channels; 51.2 GB/sec read/write	1866 MHz; 4 channels; 59.7 GB/sec read/write	2133 MHz; 4 channels; 68 GB/sec read/write
QuickPath Interconnect	4.0 GHz, 8.0 GT/s, or 32 GB/sec	4.0 GHz, 8.0 GT/s, or 32 GB/sec	4.8 GHz, 9.6 GT/s, or 38.4 GB/sec

### Inter-Node Network

	<b>Sandy Bridge</b>	<b>Ivy Bridge</b>	<b>Haswell</b>
IB Device on Node	Dual-port 4x FDR IB Mezzanine card (1 dual-port HCA); 56 Gbits/s	Dual-port 4x FDR IB Mezzanine card (1 dual-port HCA); 56 Gbits/s	Dual single-port 4x FDR IB Mezzanine card (2 single-port HCAs); 56 Gbits/s
IB Switches Between Nodes	4x FDR; 56 Gbits/s	4x FDR; 56 Gbits/s	4x FDR; 56 Gbits/s

# Preparing to Run on Pleiades Broadwell Nodes

To help you prepare for running jobs on Pleiades Broadwell compute nodes, this short review includes the general node configuration, tips on compiling your code, and PBS script examples.

## Overview of Pleiades Broadwell Nodes

Each Pleiades Broadwell rack contains 72 nodes; each node contains two 14-core E5-2680v4 (2.4 GHz) processors and 128 GB of memory, providing approximately 4.6 GB per core—slightly less than the ~5.3 GB/core provided by the Haswell nodes.

The Broadwell nodes are connected to the Pleiades InfiniBand network (ib0 and ib1) via four-lane Fourteen Data Rate (4X FDR) devices and switches for internode communication.

The home and Lustre /nobackup filesystems are accessible from the Broadwell nodes.

## Compiling Your Code For Broadwell Nodes

Like the Haswell processors, the Broadwell processors support the Advanced Vector Extensions 2 (AVX2) instructions, in addition to AVX (introduced with Sandy Bridge processors), SSE4.2 (introduced with Nehalem processors), and earlier generations of SSE.

AVX2 supports floating point fused multiply-add, integer vector instructions extended to 256-bit, and vectorization gather support, among other features. Your application may be able to take advantage of AVX2, however, not all applications can make effective use of this set of instructions. We recommend that you use the latest Intel compiler, by using the command `module load comp-intel/2020.4.304`, and experiment with the following sets of compiler options before making production runs on the Broadwell nodes:

- `-O2 -xCORE-AVX2`
- `-O3 -xCORE-AVX2`

These compiler options generate an executable that is optimized for running on Broadwell and Haswell, but cannot run on any of the older processor types. If you prefer to generate a single executable that will run on any of the existing Pleiades processor types (Sandy Bridge, Ivy Bridge, Haswell, Broadwell, Skylake, and Cascade Lake), try using one of the following sets of compiler options:

- `-O2 (or -O3)`
- `-O2 (or -O3) -axCORE-AVX512,CORE-AVX2 -xAVX`

The use of `-axCORE-AVX512,CORE-AVX2` allows the compiler to generate multiple code paths with suitable optimization to be determined at run time.

If your application does not show performance improvements with `-xCORE-AVX2` or `-axCORE-AVX512,CORE-AVX2 -xAVX` (as compared with just `-O2` or `-O3`) when running on Broadwell and Haswell nodes, it is more advantageous to use the executable built with just `-O2` or `-O3` for production runs on all Pleiades processor types.

TIP: You can add the compiler options `-ip` or `-ipo` to instruct the compiler to look for ways to better optimize and/or vectorize your code. Also, to generate a report on how well your code is vectorized, add the compiler flag `-vec-report2`.

Notes:

- If you have an MPI code, we strongly recommend that you load the `mpi-hpe/mpt` module, which always points to the [NAS recommended MPT version](#).
- Ensure your jobs run correctly on Broadwell nodes before you start production work.

## Running PBS Jobs on Broadwell Nodes

A PBS job running with a fixed number of processes or threads should use fewer Broadwell nodes than other types of Pleiades nodes, for two reasons: Broadwell nodes have more cores and more memory.

To request Broadwell nodes, use `model=bro` in your PBS script. For example:

```
#PBS -l select=xx:ncpus=yy:model=bro
```

Note: If your job requests only `model=bro` nodes, then by default PBS will run your job on either Pleiades or Electra Broadwell nodes, whichever becomes available first. If you specifically want your job to only run on Pleiades, then add `-l site=static_broadwell` to your job request. For example:

```
#PBS -l select=10:ncpus=28:mpiprocs=28:model=bro
#PBS -l site=static_broadwell
```

Similarly, a request to use Electra Broadwell nodes using `model=bro_ele` will by default run on either Pleiades or Electra Broadwell nodes. For more information, see [Preparing to Run on Electra Broadwell Nodes](#).

## Cores per Node

There are 28 cores per Broadwell node compared to 24 cores per Haswell, 20 cores per Ivy Bridge, and 16 cores per Sandy Bridge.

For example, if you have previously run a 240-process job with 10 Haswell nodes, 12 Ivy Bridge nodes, or 15 Sandy Bridge nodes, you should request 9 Broadwell nodes (where the first node can run 16 processes while the remaining 8 nodes can run 28 processes each).

### For Broadwell

```
#PBS -lselect=1:ncpus=16:mpiprocs=16:model=bro+8:ncpus=28:mpiprocs=28:model=bro
```

### For Haswell

```
#PBS -lselect=10:ncpus=24:mpiprocs=24:model=has
```

### For Ivy Bridge

```
#PBS -lselect=12:ncpus=20:mpiprocs=20:model=ivy
```

### For Sandy Bridge

```
#PBS -lselect=15:ncpus=16:mpiprocs=16:model=san
```

## Memory

Except for the Haswell node type, the Broadwell type provides more memory compared to the other Pleiades processor types both on a per node or per core basis.

For example, to run a job that needs 4 GB of memory per process, you can fit 7 processes on a 16-core Sandy Bridge node with ~30 GB/node, 15 processes on a 20-core Ivy Bridge node with ~60 GB/node, 24 processes on a 24-core Haswell node with ~122 GB/node, and 28 processes on a 28-core Broadwell node with ~122 GB/node.

Note: For all processor types, a small amount of memory per node is reserved for system usage. Therefore, the amount of memory available to a PBS job is slightly less than the total physical memory.

## Sample PBS Script For Broadwell Nodes

```
#PBS -lselect=10:ncpus=28:mpiprocs=28:model=bro
#PBS -q devel
module load comp-intel/2020.4.304 mpi-hpe/mpt
cd $PBS_O_WORKDIR
mpiexec -np 280 ./a.out
```

For more information about Broadwell nodes, see:

- [Pleiades Configuration Details](#)
- [Broadwell Processors](#)

# Preparing to Run on Pleiades Haswell Nodes

To help you prepare for running jobs on Pleiades Haswell compute nodes, this short review includes the general node configuration, tips on compiling your code, and PBS script examples.

## Overview of Haswell Nodes

Pleiades has 29 Haswell racks, each comprising 72 nodes. Each node contains two 12-core E5-2680v3 (2.5 GHz) processor chips and 128 GB of memory, providing approximately 5.3 GB of memory per core – the highest among all Pleiades processor types.

The Haswell nodes are connected to the Pleiades InfiniBand (ib0 and ib1) network via four-lane Fourteen Data Rate (4X FDR) devices and switches for inter-node communication.

The Pleiades home and Lustre (/nobackup) filesystems are accessible from the Haswell nodes.

## Compiling Your Code For Haswell Nodes

The Haswell processors support the following sets of single instruction, multiple data (SIMD) instructions:

- Advanced Vector Extensions 2 (AVX2)
- AVX (introduced with Sandy Bridge processors)
- Streaming SIMD Extensions 4.2 (SSE 4.2; introduced with Nehalem processors)
- Earlier generations of SSE

The AVX2 instruction set, introduced with Haswell processors, includes the following new features:

- Floating-point fused multiply-add (FMA) support, which can double the number of peak floating-point operations compared with those run without FMA. With 256-bit floating-point vector registers and two floating-point functional units, each capable of FMA, a Haswell core can deliver 16 floating-point operations – double the number of operations a Sandy Bridge or Ivy Bridge core can deliver.
- Integer-vector instructions support is extended from 128-bit to 256-bit capability.
- Gather vectorization support is enhanced to allow vector elements to be loaded from non-contiguous memory locations.

Your application may be able to take advantage of AVX2, however, not all applications can make effective use of this set of instructions. We recommend that you use the latest Intel compiler, through the command `module load comp-intel/2020.4.304` and experiment with the following sets of compiler options before making production runs on the Haswell nodes.

To generate an executable that is optimized for running on Haswell and Broadwell nodes (but will not run on any other Pleiades processor types), use:

```
-O2 (or -O3) -xCORE-AVX2
```

To generate a single executable that will run on any of the existing Pleiades processor types (Sandy Bridge, Ivy Bridge, Haswell, Broadwell, Skylake, and Cascade Lake), use:

- `-O2 (or -O3)`
- `-O2 (or -O3) -xCORE-AVX512,CORE-AVX -xAVX`

Note: Using **axCORE-AVX512,CORE-AVX2** allows the compiler to generate multiple code paths with suitable optimization to be determined at run time.

If your application does not show performance improvements on Haswell using either of the two AVX2 options (**-xCORE-AVX2** or **-axCORE-AVX512,CORE-AVX2 -xAVX**), as compared with just **-o2** or **-o3**, we recommend that you use the executable built with just **-o2** or **-o3** for production runs on all Pleiades processor types. Executables built with AVX or AVX2 consume more power.

If you have an MPI code that uses the HPE MPT library, use the [NAS Recommended MPT version](#), with the command:

```
module load mpi-hpe/mpt
```

TIP: If you include the **-ipo** compiler option to better optimize and/or vectorize your code, be aware that using the Intel compiler with any HPE/SGI MPT libraries can produce warning messages (regarding "**unresolved cpuset\_xxxx**" and "**bitmask\_xxx**" routines) during linking. These messages can be ignored. To make them stop, you can add the **-lcpuset -lbitmask** options.

**Important:** Ensure your jobs run correctly on Haswell nodes before starting production work.

## Running PBS Jobs on Haswell Nodes

To request Haswell nodes, use **:model=has** in your PBS script. For example:

```
#PBS -l select=xx:ncpus=yy:model=has
```

A PBS job running with a fixed number of processes or threads should use fewer Haswell nodes than the older Pleiades processor types, for the following two reasons:

- There are 24 cores per Haswell node (compared with 20 cores per Ivy Bridge, and 16 cores per Sandy Bridge node).

For example, if you have previously run a 240-process job on 12 Ivy Bridge nodes or 15 Sandy Bridge nodes, you would request 10 Haswell nodes instead:

```
For Haswell
#PBS -lselect=10:ncpus=24:mpiprocs=24:model=has
```

```
For Ivy Bridge
#PBS -lselect=12:ncpus=20:mpiprocs=20:model=ivy
```

```
For Sandy Bridge
#PBS -lselect=15:ncpus=16:mpiprocs=16:model=san
```

- Haswell processors provide more memory per core than the other Pleiades processor types. For example, to run a job that needs 5 GB of memory per process, you can fit the following number of processes on each type:
  - ◆ 24 processes on a 24-core Haswell node (or 28-core Broadwell node) with ~122 GB/node
  - ◆ 12 processes on a 20-core Ivy Bridge node with ~60 GB/node
  - ◆ 6 processes on a 16-core Sandy Bridge node with ~30 GB/node

Note: For all processor types, a small amount of memory per node is reserved for system usage. Therefore, the amount of memory available to a PBS job is slightly less than the total physical memory.

## Sample PBS Script For Haswell Nodes

```
#PBS -lselect=10:ncpus=24:mpiprocs=24:model=has
```



```
#PBS -q devel  
module load comp-intel/2020.4.304 mpi-hpe/mpt  
cd $PBS_O_WORKDIR  
mpiexec -np 240 ./a.out
```

For more detailed information about the Haswell nodes, see [Haswell Processors](#).

# Preparing to Run on Pleiades Ivy Bridge Nodes

To help you prepare for running jobs on Pleiades Ivy Bridge compute nodes, this short review includes the general node configuration, tips on compiling your code, and PBS script examples.

## Overview of Ivy Bridge Nodes

Pleiades has 75 Ivy Bridge racks, each containing 72 nodes. Each node contains two 10-core E5-2680v2 (2.8 GHz) processor chips and 64 GB of memory, providing 3.2 GB of memory per core.

The Ivy Bridge nodes are connected to the Pleiades InfiniBand (ib0 and ib1) network via the four-lane Fourteen Data Rate (4X FDR) devices and switches for inter-node communication.

The Lustre filesystems, /nobackuppX, are accessible from the Ivy Bridge nodes.

## Compiling Your Code For Ivy Bridge Nodes

Like the Sandy Bridge processor, the Ivy Bridge processor uses Advanced Vector Extensions (AVX), which is a set of instructions for doing Single Instruction Multiple Data (SIMD) operations on Intel architecture processors.

To take advantage of AVX, we recommend that you compile your code on Pleiades with an Intel compiler (version 12 or newer; use `module load comp-intel/2020.4.304` to get the latest version), using either of the following compiler flags:

- To run only on Sandy Bridge or Ivy Bridge processors: `-O2 (or -O3) -xAVX`
- To run on all Pleiades processor types (including Sandy Bridge, Ivy Bridge, Haswell, Broadwell, Skylake and Cascade Lake): `-O2 (or -O3) -xCORE-AVX512,CORE-AVX2 -xAVX`

You can also add the compiler options `-ip` or `-ipo`, which allow the compiler to look for ways to better optimize and/or vectorize your code.

To get a report on how well your code is vectorized, add the compiler flag `-vec-report2`.

If you have an MPI code that uses the HPE MPT library, use the NAS Recommended MPT version, with the command:

```
module load mpi-hpe/mpt
```

Note that mpt.2.04 and earlier versions do not support FDR.

TIP: Ensure that your jobs run correctly on Ivy Bridge nodes before starting production work.

## Running PBS Jobs on Ivy Bridge Nodes

To request Ivy Bridge nodes, use `:model=ivy` in your PBS script:

```
#PBS -l select=xx:ncpus=yy:model=ivy
```

A PBS job running with a fixed number of processes or threads should use fewer Ivy Bridge nodes than Sandy Bridge nodes for two reasons:

- There are 20 cores per Ivy Bridge node, compared with 16 cores per Sandy Bridge node.

For example, if you have previously run a 240-process job with 15 Sandy Bridge nodes, you should request 12 Ivy Bridge nodes instead.

**For Ivy Bridge**

```
#PBS -lselect=12:ncpus=20:mpiprocs=20:model=ivy
```

**For Sandy Bridge**

```
#PBS -lselect=15:ncpus=16:mpiprocs=16:model=san
```

- The Ivy Bridge processor provides more memory, per node and per core, than the Sandy Bridge processor type.

For example, to run a job that needs 2.5 GB of memory per process, you can fit the following number of processes on each type:

- ♦ 12 processes on a 16-core Sandy Bridge node with ~30 GB/node
- ♦ 20 processes on a 20-core Ivy Bridge node with ~60 GB/node

Note: For all processor types, a small amount of memory per node is reserved for system usage. Thus, the amount of memory available to a PBS job is slightly less than the total physical memory.

## Sample PBS Script For Ivy Bridge

```
#PBS -lselect=12:ncpus=20:mpiprocs=20:model=ivy
#PBS -q normal
module load comp-intel/2020.4.304 mpi-hpe/mpt
cd $PBS_O_WORKDIR
mpiexec -np 240 ./a.out
```

For more information about Ivy Bridge nodes, see:

- [Pleiades Configuration Details](#)
- [Ivy Bridge Processors](#)

# Preparing to Run on Pleiades Sandy Bridge Nodes

To help you prepare for running jobs on Pleiades Sandy Bridge compute nodes, this short review includes the general node configuration, tips on compiling your code, and PBS script examples.

## Overview of Sandy Bridge Nodes

Pleiades has 26 Sandy Bridge racks, each containing 72 nodes. Each of the 72 nodes contains two 8-core E5-2670 processor chips and has 32 GB of memory.

The E5-2670 processor speed is 2.6 GHz when Turbo Boost is OFF, and can reach 3.3 GHz when Turbo Boost is ON. When Turbo Boost is enabled, idle cores are turned off and power is channeled to the active cores, making them more efficient. The net effect is that the active cores perform above their clock speed (that is, overclocked).

The Sandy Bridge nodes are connected to the Pleiades InfiniBand (ib0 and ib1) network via the four-lane Fourteen Data Rate (4x FDR) devices and switches for inter-node communication.

The Lustre filesystems, /nobackuppX, are accessible from the Sandy Bridge nodes.

## Compiling Your Code For Sandy Bridge Nodes

One important feature of the Sandy Bridge processor is its use of the Advanced Vector Extensions (AVX), which is a set of instructions for doing Single Instruction Multiple Data (SIMD) operations on Intel architecture processors.

AVX uses 256-bit floating point registers, which are twice as wide as the 128-bit registers used in the earlier processor models on Pleiades. With two floating-point functional units and 256-bit registers (which can hold four double-precision floating-point values or eight single precision floating point values), a code with well-vectorized loops can achieve a maximum of either eight double-precision floating-point operations (flops) per cycle, per core or 16 single-precision flops per cycle, per core.

To take advantage of AVX, we recommend that you compile your code with an Intel compiler (version 12 or newer; use `module load comp-intel/2020.4.304` to get the latest version) on Pleiades, using either of the following compiler flags:

- To run only on Sandy Bridge (or Ivy Bridge) processors: `-O2 (or -O3) -xAVX`
- To run on all Pleiades processor types (including Sandy Bridge, Ivy Bridge, Haswell, Broadwell, Skylake and Cascade Lake): `-O2 (or -O3) -xCORE-AVX512,CORE-AVX2 -xAVX`

You can also add the compiler options `-ip` or `-ipo`, which allow the compiler to look for ways to better optimize and/or vectorize your code.

To get a report on how well your code is vectorized, add the compiler flag `-vec-report2`.

To compare the performance differences between using AVX and not using AVX, we recommend that you create separate executables: one with `-xAVX` or `-xCORE-AVX512,CORE-AVX2 -xAVX`, and another one without them. If you do not notice much performance improvement using these flags, then your code does not benefit from AVX.

If you have an MPI code that uses the HPE MPT library, use the `mpi-hpe/mpt` module, which always points to the NAS recommended MPT version.

Note that mpt.2.04 and earlier versions do not support FDR.

TIP: Ensure that your jobs run correctly on Sandy Bridge nodes before you start production work.

## Running PBS Jobs on Sandy Bridge Nodes

To request Sandy Bridge nodes, use `:model=san` in your PBS script:

```
#PBS -l select=xx:ncpus=yy:model=san
```

To request the devel queue, use either of the following methods:

- In your PBS script, add:

```
#PBS -q devel
```

- In your `qsub` command line, use:

```
pfe% qsub -q devel your_pbs_script
```

## Sample PBS Script For Sandy Bridge

```
#PBS -lselect=3:ncpus=16:mpiprocs=16:model=san
#PBS -q devel
```

```
module load comp-intel/2020.4.304 mpi-hpe/mpt
```


```
cd $PBS_0_WORKDIR
```

```
mpiexec -np 48 ./a.out
```

For more information about Sandy Bridge nodes, see:

- [Pleiades architecture overview](#)
- [Sandy Bridge Processors](#)

## Broadwell Processors



broadwell\_processor\_numbering.jpg

### Microarchitecture

The Intel Broadwell processor incorporated into the Pleiades cluster is the 14-core E5-2680v4 model with a clock speed of 2.4 GHz. As the 14 nanometer (nm) die shrink of the Haswell microarchitecture, the Broadwell processor uses less power and is more efficient than the Haswell processor.

### Instruction Sets

Like the Haswell processor, Broadwell supports single instruction, multiple data (SIMD) instruction sets, including several generations of Streaming SIMD Extensions (SSE, SSE2, SSE3, Supplemental SSE3, and SSE4), Advanced Encryption Standard (AES), and Advanced Vector Extensions (AVX and AVX2). Broadwell also supports some new instruction sets, including the Multi-Precision Add-Carry Instruction Extensions (ADX) for arbitrary-precision integer operations.

For information about AVX2 features and compiler support, see [Haswell Processors](#).

### Hyperthreading

Hyperthreading is turned ON.

### Turbo Boost

Turbo Boost is turned ON.

## Memory Subsystems

The memory hierarchy of Broadwell is as follows:

- L1 instruction cache: 32 KB, private to each core
- L1 data cache: 32 KB, private to each core
- L2 cache: 256 KB, private to each core
- L3 cache: 35 MB, shared by 14 cores in each socket
- Memory: 64 GB per socket, total of 128 GB per node

The Broadwell nodes are equipped with 2,400 MHz DDR4 memory to provide higher memory bandwidth. There are four memory channels per socket. Each channel can be connected with a maximum of two memory DIMMs. Of the eight memory DIMM slots for each socket, four are populated with 16-GB error correcting code (ECC)-registered DDR4 memory, for a total of 64 GB per socket. With two sockets in a node, the total memory per node is 128 GB.

Connecting the two sockets are two Intel QPI links running at a speed of 9.6 gigatransfers per second (GT/s). Each link contains separate lanes for the two directions. The total bandwidth (2 links x 2 directions) is 38.4 GB/sec.

## Network Subsystem

The network subsystem of the Broadwell nodes is the same as that of the [Haswell nodes](#), as shown in the following diagram. Each Broadwell node is equipped with two PCI Express (PCIe) interfaces (one from each socket). One PCIe interface is connected to the ib0 InfiniBand (IB) fabric via a single-port, four-lane, Fourteen Data Rate (4X FDR) host channel adapter (HCA), in a dual single-port FDR IB mezzanine card. The other PCIe interface is connected to the ib1 fabric via another single-port 4x FDR HCA in the same mezzanine card.



broadwell\_two\_ports.jpg

## Haswell Processors



haswell\_processor\_numbering.jpg

### Microarchitecture

The Intel Haswell processor incorporated into the Pleiades supercomputer is the 12-core E5-2680v3 model. Haswell uses the 22 nanometer (nm) transistors that were introduced with Ivy Bridge processor, but has a newer microarchitecture that focuses on lower power consumption and higher efficiency.

### Instruction Sets

Like the Sandy Bridge and Ivy Bridge processors, Haswell supports single instruction, multiple data (SIMD) instruction sets, including several generations of Streaming SIMD Extensions (SSE, SSE2, SSE3, Supplemental SSE3, and SSE4), Advanced Encryption Standard (AES), and Advanced Vector Extensions (AVX).

In addition, the AVX2 instruction set was introduced with Haswell processors.

### Main Features of AVX2

Features include:

- Floating-point fused multiply-add (FMA) support, which can double the number of peak floating-point operations compared with those run without FMA. With 256-bit floating-point vector registers and two floating-point functional units, each capable of FMA, a Haswell core can deliver 16 floating-point operationsâdouble the number of operations a Sandy Bridge or Ivy Bridge core can deliver.
- Integer-vector instructions support has been extended from 128-bit to 256-bit capability.



- Gather vectorization support, enhanced to enable vector elements to be loaded from non-contiguous memory locations.

## Compiler Support for AVX2

AVX2 is formally supported by Intel compilers starting with version 12.1. To take advantage of AVX2, use the following compiler and options:

### Compiler

You can use `comp-intel/2012.0.032` or later modules. We recommend using `comp-intel/2015.0.090`.

### Options

Use either `-xCORE-AVX2` or `-axCORE-AVX2`.

Both the `-xCORE-AVX2` and `-axCORE-AVX2` options turn on `-fma`, which computes  $a \times b + c$  in one rounding. This provides higher accuracy than `-no-fma`, which computes  $a \times b + c$  in two steps (first  $a \times b$ , then  $+ c$ ) and two roundings. Turn off FMA if you want your results to match legacy ones.

TIP: An application that is compiled with AVX2 instructions can run only on Haswell and Broadwell nodes. If you want a single executable that will run on any of the Pleiades processor types, with suitable optimization to be determined at run time, you can compile your application using the option `-O3 -ipo -axCORE-AVX2 -xSSE4.2`.

For Fortran codes, consider using the option `-align array32byte` to align your vector arrays on 32-byte boundaries for best performance and reliability. This option is available in Intel compiler version 13 (for example, `comp-intel/2013.1.117` and later).

AVX2 is also supported in the GNU Compiler Collection (GCC) starting with version 4.7. Use compiler options such as `-march=core-avx-2` or `-mavx2`, `-mfma` if you want to use AVX2.

## Hyperthreading

Hyperthreading is turned ON.

## Turbo Boost

Turbo Boost is turned ON.

## Memory Subsystems

The memory hierarchy of Haswell is as follows:

- L1 instruction cache: 32 KB, private to each core
- L1 data cache: 32 KB, private to each core
- L2 cache: 256 KB, private to each core
- L3 cache: 30 MB, shared by 12 cores in each socket
- Memory: 64 GB per socket, total of 128 GB per node

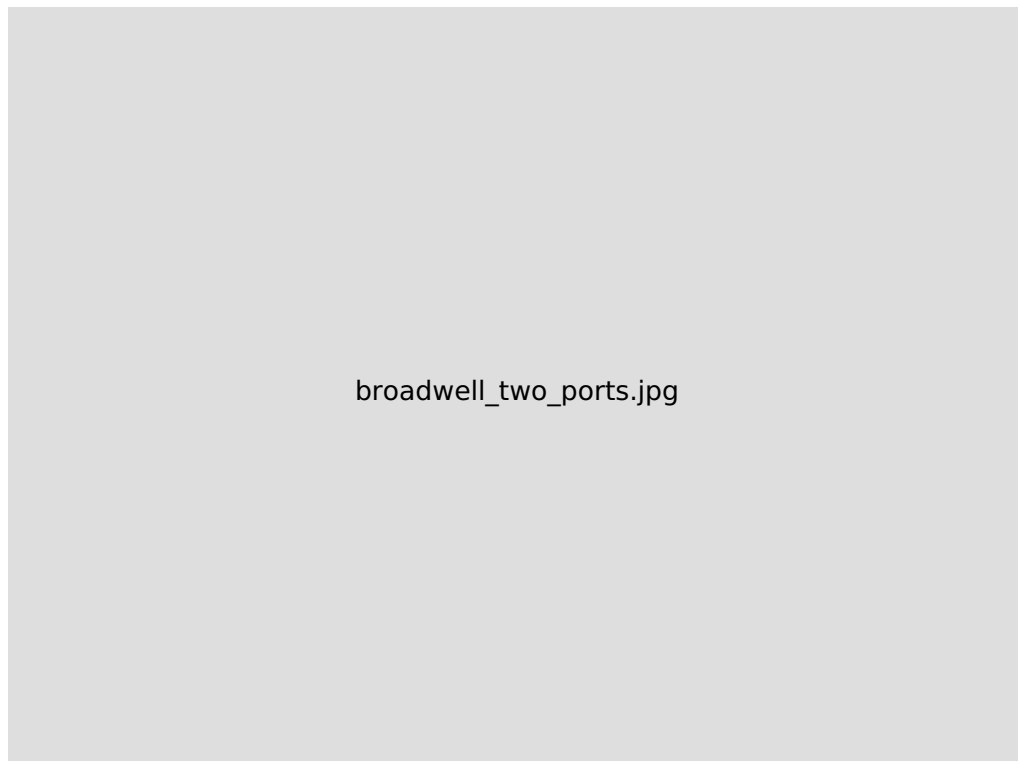
The Haswell nodes are equipped with higher speed (2,133 MHz) DDR4 memory to provide higher memory bandwidth. There are four 2,133 MHz memory channels per socket. Each channel can be connected with up to two memory DIMMs. Of the eight memory DIMM slots for each socket, four are populated with 16-GB error correcting code (ECC)-registered DDR4 memory, for a total of 64 GB per socket. With two sockets in a node, the total memory per node is 128 GB.

Connecting the two sockets are two Intel QPI links running at a speed of 9.6 gigatransfers per second (GT/s) . Each link contains separate lanes for the two directions. The total bandwidth (2 links x 2 directions) is 38.4 GB/sec.

## Network Subsystem

The Haswell and Broadwell nodes are connected to the two fabrics (ib0 and ib1) of the Pleiades InfiniBand (IB) network in a different way from the earlier processor nodes, as shown in the diagram below. Note the following differences:

- In Haswell and Broadwell node connections, each of the two PCI Express Interfaces—one from each socket—is connected to a separate single-port, four-lane, Fourteen Data Rate (4X FDR) host channel adapter (HCA), in a dual single-port FDR IB mezzanine card.
- In Sandy Bridge and Ivy Bridge node connections, only one of the two PCI Express Interfaces is connected to a dual-port FDR IB HCA, in a dual-port FDR IB mezzanine card.



The IB mezzanine card sits on a "sister board" next to the motherboard on each node. The motherboard contains the two processor sockets. There are 18 nodes per individual rack unit. To join the ib0 fabric, the nodes are connected to two Mellanox FDR IB switches, in an ICE X IB Premium Blade. To join the ib1 fabric, another set of connections between the 18 nodes and a second Premium Blade is established.

## Ivy Bridge Processors



ivybridge\_processor\_numbering.jpg

### Core Labeling

The core labeling in Ivy Bridge is contiguous. That is, cores 0-9 are in the first socket and cores 10-19 are in the second socket.

When using the HPE MPT library, the environment variable `MPI_DSM_DISTRIBUTE` is set to ON by default for the Ivy Bridge nodes.

### Instruction Set

The Ivy Bridge processor is a die shrink (22 nm) of the Sandy Bridge processor (32 nm). It uses the Sandy Bridge micro-architecture, which contains Advanced Vector Extensions (AVX), a set of instructions for doing Single Instruction Multiple Data (SIMD) operations. These extensions widen the vector registers from 128 bits to 256 bits, so the floating-point hardware can sustain 16 single-precision or 8 double-precision floating point operations per cycle. As a result, even though the CPU clock speed of the Ivy Bridge processor (2.8 GHz) is lower than that of the earlier processor models on Pleiades, the floating-point performance can be higher for some applications.

AVX is formally supported in Intel compilers starting with version 12. Use the `comp-intel/2012.0.032` or newer modules (as of November 2014, `comp-intel/2015.0.090` is recommended) to take advantage of AVX. For Fortran codes, consider using the option `-align array32byte` (available in Intel compiler version 13 and above) to align your vector arrays on 32-byte boundaries for best performance and reliability.

AVX is also supported in the GNU Compiler Collection starting with version 4.6. An application that is compiled with AVX instructions can run only on Sandy Bridge or Ivy Bridge.

TIP: If you want a single executable that will run on any of the Pleiades processor types, with suitable optimization to be determined at run time, you can compile your application using the option `-O3 -ipo -axCORE-AVX2 -xSSE4.2`.

## Hyperthreading

Hyperthreading is turned ON.

## Turbo Boost

Turbo Boost is turned ON.

## Memory Subsystems

The memory hierarchy of Ivy Bridge is as follows:

- L1 instruction cache: 32 KB, private to each core
- L1 data cache: 32 KB, private to each core
- L2 cache: 256 KB, private to each core
- L3 cache: 25 MB, shared by 10 cores in each socket
- Memory: 32 GB per socket, total of 64 GB per node

There are four 1866-MHz memory channels per socket. Each channel can be connected with up to two memory DIMMs. Of the eight memory DIMM slots for each socket, four are populated with 8-GB Error Correcting Code (ECC) registered DDR3 memory, for a total of 32 GB per socket. With two sockets in a node, the total memory per node is 64 GB.

Connecting the two sockets are two Intel QPI links running at a speed of 8.0 Giga-transfers (GT) per second. Each link contains separate lanes for the two directions. The total bandwidth (2 links x 2 directions) is 32 GB/sec.


## Network Subsystem

The Ivy Bridge nodes are connected to the two fabrics (ib0 and ib1) of the Pleiades InfiniBand (IB) network via a dual-port, four-lane Fourteen Data Rate (4X FDR) IB Mezzanine card on each node, as well as the Mellanox FDR IB switches in the ICE X IB Premium Blade. The FDR runs at 14 Gb/sec per lane. With four lanes, the total bandwidth is 56 Gb/sec or about 7 GB/sec.

On each node, the IB Mezzanine card sits on a sister board next to the mother board, which contains the two processor sockets.

There are 18 nodes per Individual Rack Unit. These 18 nodes are connected to two Mellanox FDR IB switches in an ICE X IB Premium Blade to join the ib0 fabric. Another set of connections between the 18 nodes and a second Premium Blade is established for ib1.

## Sandy Bridge Processors



sandybridge\_processor\_numbering.jpg

### Core Labeling

The core labeling in Sandy Bridge is contiguous. That is, cores 0-7 are in the first socket and cores 8-15 are in the second socket.

When using the HPE MPT library, the environment variable **MPI\_DSM\_DISTRIBUTE** is set to ON by default for the Sandy Bridge nodes.

### Instruction Set

A Sandy Bridge processor's execution hardware contains the Advanced Vector Extensions (AVX), a set of instructions for doing Single Instruction Multiple Data (SIMD) operations on Intel architecture processors. These extensions widen the vector registers from 128 bits to 256 bits, so the floating-point hardware can sustain 16 single-precision and 8 double-precision floating point operations per cycle. As a result, even though the CPU clock speed of the Sandy Bridge processor (2.6 GHz) is lower than that of older processor types on Pleiades, the floating-point performance can be higher for some applications.

AVX is supported in Intel compilers starting with version 11.1. However, Intel versions 12 and 13 compilers provide more optimizations for AVX and are recommended over version 11.1.

AVX is also supported in the GNU Compiler Collection starting with version 4.6. An application that is compiled with **-xAVX** can run on Sandy Bridge or Ivy Bridge.

**TIP:** If you want to have a single executable that will run on any of the Pleiades processor types, with suitable optimization to be determined at run time, you can compile your application with:  
**-O3 -ipo -xCORE-AVX2 -xSSE4.2.**

## **Hyperthreading**

Hyperthreading is turned ON.

## **Turbo Boost**

Turbo Boost is turned ON.

## **Memory Subsystems**

The memory hierarchy of Sandy Bridge is as follows:

- L1 instruction cache: 32 KB, private to each core
- L1 data cache: 32 KB, private to each core
- L2 cache: 256 KB, private to each core
- L3 cache: 20 MB, shared by 8 cores in each socket
- Memory: 16 GB per socket, total of 32 GB per node

There are four 1600-MHz memory channels per socket. Each channel can be connected with up to two memory DIMMs. Of the eight memory DIMM slots for each socket, four are populated with 4-GB Error Correcting Code (ECC) registered DDR3 memory, for a total of 16 GB per socket. With two sockets in a node, the total memory per node is 32 GB. If there is a user requirement, some nodes could be configured with larger amounts of memory.

Connecting the two sockets are two Intel QPI links running at a speed of 8.0 Giga-transfers (GT) per second. Each link contains separate lanes for the two directions. The total bandwidth (2 links x 2 directions) is 32 GB/sec.

## **Network Subsystem**

The Sandy Bridge nodes are connected to the two fabrics (ib0 and ib1) of the Pleiades InfiniBand (IB) network via the dual-port, four-lane Fourteen Data Rate (4x FDR) IB Mezzanine card on each node, as well as the Mellanox FDR IB switches in the ICE X IB Premium Blade. The FDR runs at 14 Gb/sec per lane. With four lanes, the total bandwidth is 56 Gb/sec or about 7 GB/sec.

On each node, the IB Mezzanine card sits on a sister board next to the mother board, which contains the two processor sockets.

There are 18 nodes per Individual Rack Unit. These 18 nodes are connected to two Mellanox FDR IB switches in an ICE X IB Premium Blade to join the ib0 fabric. Another set of connection between the 18 nodes and a second Premium Blade is established for ib1.

## Hyperthreading

Hyperthreading is available and enabled on the Pleiades, Aitken, and Electra compute nodes. With hyperthreading, each physical core can function as two logical processors. This means that the operating system can assign two threads per core by assigning one thread to each logical processor.

Note: Hyperthreading is currently off on Aitken Rome nodes.

The following table shows the number of physical cores and potential logical processors available for each processor type:

Processor Model	Physical Cores (N)	Logical Processors (2N)
<u>Sandy Bridge</u>	16	32
<u>Ivy Bridge</u>	20	40
<u>Haswell</u>	24	48
<u>Broadwell</u>	28	56
<u>Skylake</u>	40	80
<u>Cascade Lake</u>	40	80

If you use hyperthreading, you can run an MPI code using  $2N$  processes per node instead of  $N$  process per node—so you can use half the number of nodes for your job. Each process will be assigned to run on one logical processor; in reality, two processes are running on the same physical core.

Running two processes per core can take less than twice the wall-clock time compared to running only one process per core—if one process does not keep the functional units in the core busy, and can share the resources in the core with another process.

## Benefits and Drawbacks

Using hyperthreading can improve the overall throughput of your jobs, potentially saving standard billing unit (SBU) charges. Also, requesting half the usual number of nodes may allow your job to start running sooner—an added benefit when the systems are loaded with many jobs. However, using hyperthreading may not always result in better performance.

**WARNING:** Hyperthreading does not benefit all applications. Also, some applications may show improvement with some process counts but not with others, and there may be other unforeseen issues. Therefore, before using this technology in your production run, you should test your applications with and without hyperthreading. If your application runs more than two times slower with hyperthreading than without, do not use it.

## Using Hyperthreading

Hyperthreading can improve the overall throughput, as demonstrated in the following example.

### Example

Consider the following scenario with a job that uses 40 MPI ranks on Ivy Bridge. Without hyperthreading, we would specify:

```
#PBS -lselect=2:ncpus=20:mpiprocs=20:model=ivy
```

and the job will use 2 nodes with 20 processes per node. Suppose that the job takes 1000 seconds when run this way. If we run the job with hyperthreading, e.g.:

```
#PBS -lselect=1:ncpus=20:mpiprocs=40:model=ivy
```

then the job will use 1 node with all 40 processes running on that node. Suppose this job takes 1800 seconds to complete.

Without hyperthreading, we used 2 nodes for 1000 seconds (a total of 2000 node-seconds); with hyperthreading, we used 1 node for 1800 seconds (1800 node-seconds). Thus, under these circumstances, if you were interested in getting the best wall-clock time performance for a single job, you would use two nodes without hyperthreading. However, if you were interested in minimizing resource usage, especially with multiple jobs running simultaneously, using hyperthreading would save you 10% in SBU charges.

## Mapping of Physical Core IDs and Logical Processor IDs

Mapping between the physical core IDs and the logical processor IDs is summarized in the following table. The value of  $N$  is 16, 20, 24, 28, and 40 for Sandy Bridge, Ivy Bridge, Haswell, Broadwell, and Skylake/Cascade Lake processor types, respectively.

Physical ID	Physical Core ID	Logical Processor ID
0	0	0 ; $N$
0	1	1 ; $N+1$
...	...	.....
0	$N/2 - 1$	$N/2 - 1$ ; $N + N/2 - 1$
1	$N/2$	$N/2$ ; $N + N/2$
1	$N/2 + 1$	$N/2 + 1$ ; $N + N/2 + 1$
1	...	...
1	$N - 1$	$N - 1$ ; $2N - 1$

Note: For additional mapping details, see the configuration diagrams at the for each processor type, or run the `cat /proc/cpuinfo` command on the specific node type.